

GREATER AVAILABLE PROCESSING POWER AND BETTER ALGORITHMS CONTINUE TO MAKE SPEECH RECOGNITION MORE VIABLE FOR MANY REAL-WORLD APPLICATIONS. THE REAL ADVANCES, HOWEVER, HAVE BEEN IN HOW YOU CAN USE SPEECH TO ACHIEVE "NATURAL UNDERSTANDING."

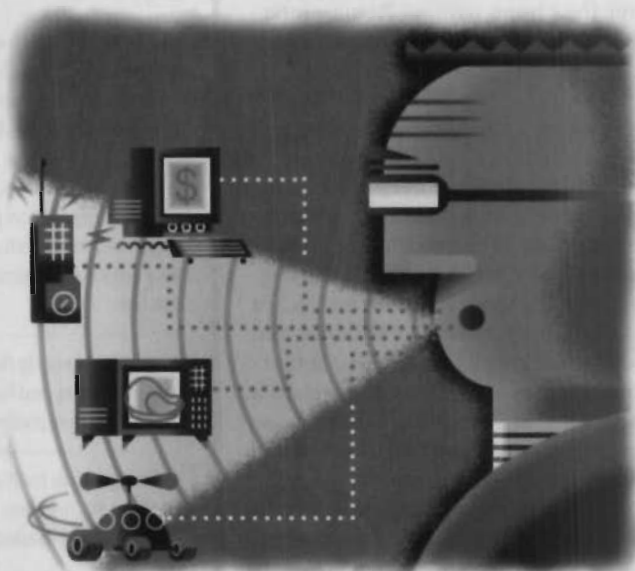


Illustration by Daniel Guidera

SPEECH RECOGNITION: **It's not what you say;** *it's how you say it*

SPEECH APPLICATIONS THAT DIRECT YOU TO "Say 1 if you want to do x," miss the true power of speech. Even applications that walk through a script—sometimes known as "phone jail"—can leave you wondering where the dialogue is going and whether you can get to your desired goal. Most people would rather get straight to the point and say, "Call home" or "Buy 100 shares of NCC at

\$4.50." Fortunately, speech-recognition technology can now let you use devices in a way you already understand—by talking—instead of requiring you to use an unfamiliar interface. The key to getting the most from speech recognition lies in taking advantage of developments in interface design that speech recognition enables.

Today's DSPs can provide enough processing power to recognize speech from a reasonably sized vocabulary with sufficient accuracy. The two killer applications for speech recognition this year are hands-free adapters for car phones, which use low-end command-and-control technology, and automated directory assistance, which uses high-

At a glance	80
Optimizing recognition performance and increasing accuracy	80
Speaker-independent or -dependent templates	82
The other half of the conversation: speech synthesis	84
Knows his master's voice	86
For more information	88



end natural-understanding technology. Soon, you'll be able to buy and sell stock over the phone without human intervention. Even toys and simple appliances can use speech recognition.

The most recent advances in speech recognition come through context and natural-language understanding. In "mixed-initiative" mode, either the speaker or the voice system can direct the conversation. With modeless or conversational systems, either side can say "anything" at any time. Users are not locked into a script; phrasing, word choice, and word order are all flexible. One important feature of a conversational system is the ability to "bargain in," allowing the speaker to interrupt the voice system with either an answer to the question the system is currently asking or a command directing action in another context. For example, during an exchange to buy stock, the speaker could ask an out-of-context question, such as the price of a different stock.

Where can you implement speech-recognition technology? Because recog-

AT A GLANCE

▷ You can find speech-recognition technology in almost every form factor, from dedicated ICs to portable C algorithms for DSPs to turnkey boards for telephony systems.

▷ Because recognition engines turn speech into text, speech can serve as a front end for any application with a text interface.

▷ Adding memory to the recognition engine can often yield faster recognition than adding more processing power.

▷ Dialogue tools build conversational interfaces that free users from constrained scripts and allow speakers to speak more naturally.

nition engines turn speech into text, speech recognition can serve as a front end to any system supporting a text in-

terface. You can implement the speech-recognition engine in several ways. For simple to complex applications, you can buy speech-recognition-specific ICs, run a software engine on a DSP, or locate speech-recognition processing on a host server. Voice recognition has even penetrated the Web with VXML (voice-extendible markup language; www.vxmlforum.org).

COMMAND AND CONTROL

For low-end applications requiring limited vocabulary, you can buy speech-recognition-specific ICs such as Sensory's RSC-264T and RSC-364 speech-recognition μ Cs with ADCs and DACs. With as much as 64 kbytes of ROM and 2.5 kbytes of RAM, the chips can support a combination of technologies, including a speaker-independent engine, a speaker-dependent engine, speaker adaptation, speech verification, voice recording, and speech and music synthesis; each module requires 5 to 10 kbytes of memory. Speaker-dependent templates require 100 bytes per 3 sec of speech, speak-

OPTIMIZING RECOGNITION PERFORMANCE AND INCREASING ACCURACY

- For short vocabulary sets, use dissimilar-sounding words such as "hat," "kitten," and "mouse" instead of "hat," "cat," and "rat" to avoid word confusion. You can also vary the number of syllables. For example, use "orange," "watermelon," and "grape" instead of "orange," "apple," and "cherry."

- In general, sentences are easier to recognize than words, given that a sentence has more variation from other sentences than words do from words. Longer responses, such as "Buy stocks" or "View my portfolio," are easier to recognize than shorter ones, such as "Buy" or "View."

- Dynamically reducing vocabulary based on the current context increases accuracy. For example, if the user must supply a dollar amount, the system need not search most of the vocabulary because the speaker will not use it.

- Record training samples in a typical target environment. This step includes using the same microphone; ambient noise; a closed environment, such as a car; distance of the microphone from the speaker; and other parameters. You should also duplicate the state of a typical user, both physically and emotionally. (For example, people using exercise equipment with speech recognition tend to breathe heavily, office users are quiet and calm, and children get loud and excited.)

- Build questions that suggest available responses. For example, "What do you want to do?" is too open a question. "Who do you want to call?" leads the user to a response that the application can understand.

- Note that, although "bargain in" is a desirable feature, it requires a full-duplex interface and therefore processor-intensive echo cancellation.

- Structure your dialogue to clarify vocabulary and put it into smaller subsets. For example, in a telephone-directory application, instead of having the user ask for John Smith and then for John's department to locate the right John Smith, first determine the department. By doing so, the engine must recognize a department from a small subset of the vocabulary and then can reduce the search space for the name from the entire directory to only those people in the department.

- Discrete speech-recognition engines require the user to pause between each word or digit. Such pauses are unnatural and can even cause users to talk artificially or monotonically, reducing accuracy. Continuous engines require more processing power but present a better interface.

- If the speech-recognition engine does not understand an answer the first time, it will probably not understand it a second

time. Restructure clarification questions, perhaps asking for a different format ("sixty-four" versus "six, four"). In any case, some users will innocently confound the system. You must be able to identify these users and handle their needs, either by switching to a less ambiguous interface, such as "Press 1 to do x," or by routing the users to a person.

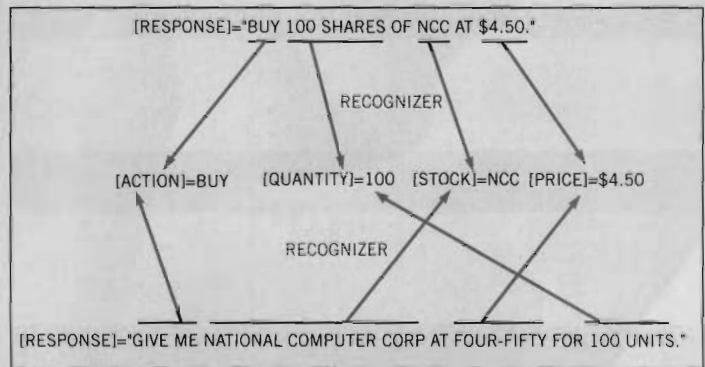
- The ability to record a user's voice is important for building a conversational "confirmation structure." With a phone, for example, the name "Conrad" may mean a particular phone number. When the user trains the phone to understand "Conrad," the phone could record the user's voice saying, "Conrad" instead of reading back Conrad's number. In this way, when the speech-recognition engine doesn't recognize the name "Conrad," the phone could ask, "Are you trying to call Conrad?"

er-independent templates require 5 kbytes for 14 words, and synthesis requires 1 to 2 kbytes per sec of speech or music (see sidebar "Speaker-independent or -dependent templates"). The μ C can also serve as a general-purpose μ C to control lights or motors. Sensory offers a development kit, including a chip simulator, a compiler, a linker, code samples, development boards, and a tutorial for \$3200. You can also purchase the chip as a module, and the various algorithms are available in C for use in DSPs.

As inexpensive as Sensory's chips are—\$2.25 (100,000) for the RSC-264T—you'll still find it less expensive to run speech-recognition software on a DSP if you have one available. Also, to perform operations such as acoustic echo cancellation, you need a DSP. However, you may find the time-to-market benefits of adding a chip versus integrating software worth the extra cost.

Several DSP companies, such as Analog Devices and Philips, offer DSPs with ADCs and DACs supporting half- or full-duplex channels and various memories, including flash. Depending on the performance of the DSP and available memory, you can implement a variety of filters, speech-recognition engines, and vocabulary sizes. Because these DSP chips are programmable, you can update algorithms and vocabularies. Be sure to check with the chip companies' partners to see which algorithms they have already ported to the chips. Something else to watch out for: Some of these chips are really multichip modules, meaning that several pieces of silicon come in the same package. By putting these pieces—the

Figure 1



Parsing voice responses into semantic slots extracts meaning from complex utterances. Responses with equivalent meaning yield equivalent semantic breakdowns, regardless of word order or choice of aliases.

DSP, converters, and memory—together yourself, you can choose the optimal balance and cost for your application.

On the board side, companies such as Radisys offer platforms for supporting voice recognition, from call centers to dictation centers that have 500 incoming lines. THB Germany offers embeddable modules for hands-free adapters for car phones. Many board companies supply no speech-recognition software but partner with companies that do. Some boards offer application-specific functions, such as the ability to decompress compressed voice (voice-over-Internet Protocol applications), handle protocols, and even route calls. In a server environment, you can run a powerful speech-recognition engine on the server, using a front-end engine to extract speech features to send to the server for recognition. In this way, the cost of speech-recognition process-

ing falls on the server, where many nodes can take advantage of the benefits.

Given that speech recognition is technically feasible for a variety of applications, vocabulary size is the key factor to consider in defining the characteristics of a speech-recognition engine. If the engine has a vocabulary of only 10 words, the engine has a high probability of recognizing the right word. Such an engine probably doesn't have to work that hard—it's easy to distinguish between "yes" and "no." The larger the vocabulary, however, the more robust the engine you need to distinguish between similar-sounding words, such as "Austin" and "Boston," which decrease confidence scores—that is, how confident the engine is that it accurately recognizes the word—and increase recognition time (see sidebar "Optimizing recognition performance and increasing accuracy").

SPEAKER-INDEPENDENT OR -DEPENDENT TEMPLATES

An independent speech-recognition engine can, in theory, understand the speech of all but the most difficult speakers with accents, although you can adjust an engine for regional characteristics. Dependent engines, on the other hand, learn a person's voice and the nuance of the person's speech, and when a system builds a template from this person, the system can use that template only with this person.

Applications such as cell phones may employ both technologies. These applications may use speaker-independent technology for standard commands, such as "call," and speaker-dependent technology for learned commands, such as "home" or "office." Each user of a dependent system needs to enroll. Enrollment can be as simple as a user saying a recognizable word, saying a few words to

determine which of x standard templates is best for this user, reading a script to provide enough data to build a template, or even supporting correction of inaccurately recognized words.

You are not limited to one speech template. For example, you can dynamically select a template based on the initial statement that a person makes, enabling the speech-recognition engine to identify a regional

accent and use a region template better suited to the speaker. A system could also create templates for each user for applications that have multiple users. Each template stores an adaptive model of the user's voice. Either the user could identify himself or herself directly, or the engine could automatically recognize the user as user 1 or user 2, with user 0 reserved for unregistered users.

Vocabulary size also determines how much memory you need. You can store vocabularies in ROM if you do not expect them to change or if you are not using a dynamic model that swaps portions of the vocabulary into and out of memory as needed. For telephony systems, the larger the vocabulary, the more MIPS you need to process speech, and the fewer the channels you can support.

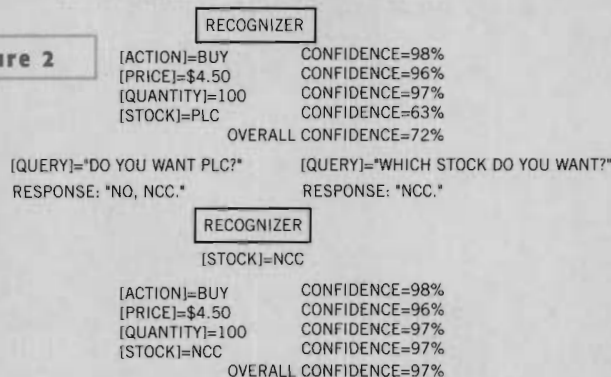
If you need fast recognition, don't be afraid to throw memory at your problem. Memory is often more important than processor speed. Consider a case in which only half of the vocabulary (the half used most frequently) is available in local memory. If

the speaker uses words from the other half of the vocabulary, loading sections of the second half may result in a longer delay than a search of the entire vocabulary would take if it was readily available.

Note that building a vocabulary, even a small one, is not a trivial undertaking. Companies building speaker-independent models commonly take 500 samples for each word. You can find most common words, such as "yes" and "no," off the shelf. With numbers, consider supporting more than just zero through nine, especially if you want to say dollar values (\$22.63) or even phone numbers (for example, I say the last four digits of my

[RESPONSE]="BUY 100 SHARES OF NCC AT \$4.50."

Figure 2



In single-phase correction, a 72% confidence score for a complete response is too low to assume that the system has understood the response. The low confidence of [stock], however, brings down the overall score. Confirming or asking the stock choice again increases the confidence of [stock] and thus raises the overall confidence to a level sufficient for the system to act. Note that, generally, each semantic slot has several recognized meanings, listed by declining confidence. The grammar selects the most likely meaning based on confidence and context.

phone number as "eighty-nine-oh-six" for 8906). If you need custom words, your speech-recognition supplier can probably build models for you, but remember that such services can take several months.

DIALOGUE TOOLS

Even more important than the size of your vocabulary is how you use that vocabulary. You can buy a turnkey speech-recognition system, complete with boards, running code, application-programming interfaces, drivers, and vocabularies, but you have to face what may be the most challenging phase of design: creating the interface dialogue.

When the recognition engine converts speech to text, the engine often passes on several possible text answers, each with a confidence rating. Now, natural understanding does not imply complex artificial intelligence; the engine doesn't "understand" what a speaker is saying. Rather, dialogue tools generate grammars, which attempt to validate the possible answers based on context; select the most likely response; and then map this response to a direct action, such as dialing a phone.

Dialogue tools, such as those from Unisys and many of the companies offering speech-recognition engines, effectively build state machines. From any state in the potential conversation, control can transfer to another state based on certain input values. Unfortunately, most of these tools are still text-based and do not offer a graphical-user-interface, high-level view of the constructed dialogue. Thus, visualizing the myriad paths a conversation can take can be difficult.

Good dialogue tools help speed the development of a system and allow you to perform a dry run on a system with voice files, letting test users actually talk to the system. For custom systems, your customers can directly experience the "listen and feel." Currently, the available dialogue tools do not autocode, but the models are straightforward to modify

THE OTHER HALF OF THE CONVERSATION: SPEECH SYNTHESIS

Fundamental to building a useful speech-recognition system is the ability to communicate back to the user. Realistically, there is no such thing as 100%-accurate recognition. Even between two people, confusion occasionally arises in a conversation. Additionally, sometimes your tongues slip, and you say completely different words from those you intended. For times like these, you must be able to requery the user.

Certainly, you can flash a light or beep to indicate an error, but

the most natural way to communicate an error in response to speech is with speech. Additionally, if speech recognition frees the hands, speech synthesis frees the eyes. A doctor, for example, may be unable to look away during a procedure. In such a situation, the doctor could ask an instrument with speech synthesis what the instrument's status is.

Two kinds of speech synthesis exist: prerecorded messages and real-time, text-to-speech conversion. Today, recorded speech

sounds better than synthesized speech because the human whose voice is recorded knows the context of the message and can add appropriate emphasis. Synthesizers face the challenge of naturally stitching words together, although stitching recorded words can give you lower quality than synthesized stitching because synthesizers can make some contextual adjustments.

Companies such as Elan and Acuvoice have applied many resources to generating synthe-

sized speech (text to speech), which sounds more like a person than a machine. Some of these companies' achievements include the addition of inflection and nuances in pronunciation, context (reading a date as a date rather than as numbers and slashes), appropriate pauses, personality in the voice, and others. Fluent Speech even offers a graphical head that mouths words the way a person would. Note that speech synthesis is not standard in most recognition packages.



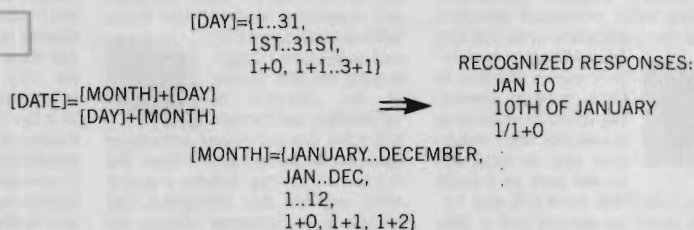
and provide a detailed specification for the development team. (Autocoding is coming soon.)

Building dialogues is not easy if you want to work on a natural-understanding level. For example, there are easily 50 synonyms or aliases for saying "yes." Although you could force a user to use "yes," doing so then requires that a user spend time learning the correct vocabulary for your system. In applications in which a user may call in only once, training is not an option. In any case, supporting reasonable aliases makes your system easier to use and gives it a competitive edge over rigid systems that require set responses.

OBJECTS IN SPEECH

You can also parse spoken sentences into semantic slots (Figure 1). For example, a simple stock transaction requires several pieces of information: the action, the quantity, the stock, and the price. These semantic slots can occur in a variety of orders. The grammar—the states, prompts, responses, and vocabulary that the dialogue tool creates—knows that all four pieces of information are necessary for a complete command. One advantage of using semantic slots is single-phrase correction (Figure 2). For example, if the confidence of the complex phrase, "Buy 100 shares of NCC at \$4.50," is low, one of the semantic slots, such as NCC, may be low and thus drops the overall score. The grammar engine can evaluate that the other semantic slots have a high confidence and that only the [stock] slot is in question. The grammar could then ask, "Which stock?" without making the user

Figure 3



Semantic slots are objects you can use in sentences of the grammar to represent many responses without enumerating each possibility. Slots may contain other slots. For example, [DATE] uses [DAY] and [MONTH]. Note that 1 + 0 represents distinctly speaking each digit: "one, zero" for 10.

Figure 4

GRAMMAR: (RECOGNIZED RESPONSE $\Rightarrow$ ACTION)
CALL [LIST] $\Rightarrow$ CALL THE NUMBER ASSOCIATED WITH THE RECOGNIZED NAME
CHANGE NUMBER FOR [LIST] $\Rightarrow$ EDIT THE NUMBER ASSOCIATED WITH THE RECOGNIZED NAME
WHO IS [LIST] $\Rightarrow$ READ OUT THE NUMBER ASSOCIATED WITH THE RECOGNIZED NAME

[PHONE LIST 1]=[OFFICE, 1-510-558-8906;
CONRAD, 1-510-558-8914]

[PHONE LIST 2]=[OFFICE, 1-617-767-2676;
MARTA, 1-617-767-2677]

[LIST]=[PHONE LIST 1] $\Rightarrow$ *CALL OFFICE* $\Rightarrow$ DIAL 1-510-558-8906
CALL CONRAD $\Rightarrow$ DIAL 1-510-558-8914
CHANGE NUMBER FOR OFFICE $\Rightarrow$ EDIT 1-510-558-8906
CHANGE NUMBER FOR CONRAD $\Rightarrow$ EDIT 1-510-558-8914
WHO IS OFFICE? $\Rightarrow$ READ OUT 1-510-558-8906
WHO IS CONRAD? $\Rightarrow$ READ OUT 1-510-558-8914

[LIST]=[PHONE LIST 2] $\Rightarrow$ *CALL OFFICE* $\Rightarrow$ DIAL 1-617-767-2676
CALL MARTA $\Rightarrow$ DIAL 1-617-767-2677
CHANGE NUMBER FOR OFFICE $\Rightarrow$ EDIT 1-617-767-2676
CHANGE NUMBER FOR MARTA $\Rightarrow$ EDIT 1-617-767-2677
WHO IS OFFICE? $\Rightarrow$ READ OUT 1-617-767-2676
WHO IS MARTA? $\Rightarrow$ READ OUT 1-617-767-2677

By defining the grammar with the object [list], you can change contexts—in this case, the current user—without having to redesign a whole new grammar.

repeat the entire command.

Semantic slots also reduce the work you need to do to design a dialogue. Instead of having to define a sentence for the grammar for each possible response, you can use semantic slots. For example, instead of "Say 1 to call mom," "Say 2 to call home," and so on, you can use "Say

1 to call [name]." Semantic slots are objects that you can use in other sentences of the grammar and can include objects themselves (Figure 3). Slots can also ease the challenge of creating dynamic grammars based on different contexts. For example, you can personalize a grammar by using different phone lists; in either case,

KNOWS HIS MASTER'S VOICE

Companies such as Voice Control Systems offer voice verification with their recognition engines, so these system both recognize and verify identities. Several camps argue the reliability of voice authentication. If users identify themselves over a less-than-perfect source, such as a telephone, the sample to verify is low-quality. Such verification systems are easy to fool using recorded responses. One

method for stalling such attacks is to require users to answer random prompts that are difficult to anticipate and prerecord. Unless you have a powerful voice engine linked with other reliable biometrics, you might consider speaker verification about as secure as a car lock: Anyone who wants to can break in. This fact is not to say that verification is without value. It keeps honest people honest: Locking your cell

phone or digital diary with a verbal password keeps out the casual snoop.

Verification focuses on false acceptance and false rejection. If you care about security, you want to false-reject users more often than you false-accept them. However, beware of falling into the trap of designing the "perfect" system at the expense of ease of use. By designing a system to account

for every sticky possibility, you may make your system difficult for the average person to use; if a system keeps rejecting a user, that user will stop using the system. In many cases, convenience is a key factor in using voice, and you may be better off relying on an alternative form of verification, such as personal-identification numbers or passwords.



the form and structure of the prompts and responses stay the same, based on the [list] slot (Figure 4).

GETTING THE RIGHT BALANCE

The trick to building a good dialogue is finding the right balance. Allowing users to say too much without direction can scare them, but constricting options can be frustrating. Also note that, although dialogue tools seem best suited for complex applications, many of the same fundamentals apply to low-end, command-and-control applications. An engine supporting a vocabulary of 100 words probably doesn't need to recognize variations of sentences. However, the issues of aliases, context, conversational interface, and craftsmanship of the complete grammar apply just as—if not even more—strongly because you have less

room to make sure you've covered all the ways people can respond. The best interfaces reduce, not increase, user errors, and the true measure of a successful implementation is achieving invisibility to the user.

Unfortunately, desktop dictation systems, such as IBM's ViaVoice and Dragon's Naturally Speaking have created mixed consumer expectations for speech-recognition technology. (For a description of my challenges to write portions of this article with speech-recognition technology, check out the electronic version of this article at www.edn-mag.com.) Dictation is the hardest of all speech-recognition applications in that, often, no context exists; sentences can be about any subject and can use

any word in the vocabulary. As a consequence, inaccuracies—95% accuracy results in one error in every 20 words—are confusing, frustrating, and annoying, and they tend to reduce user expectations.

For nondictation applications, however, the future sounds exciting. Companies that used to offer only turnkey products have started to sell their low-level technology. Chip companies often offer a software version of the algorithm, and boards and dialogue tools can work with a variety of speech-recognition engines. This situation is good news; it means that speech recognition is shifting from a "complete-solution" industry to one offering best-of-breed components. □

You can reach
Technical Editor
Nicholas Cravotta at
1-510-558-8906,
fax 1-510-558-8914,
e-mail cravotta@compuserve.com.



FOR MORE INFORMATION...

For more information on products such as those discussed in this article, circle the appropriate numbers on the Information Retrieval Service card or use EDN's InfoAccess service. When you contact any of the following manufacturers directly, please let them know you read about their products in EDN.

Acuvoice, a Fonix Co

1-408-861-1300
www.acuvoice.com
Circle No. 314

Advanced Recognition Technologies

1-805-581-3999
www.artcomp.com
Circle No. 315

Analog Devices

1-781-937-1428
www.analog.com
Circle No. 316

Babel Technologies

+ 32 65 37 42 75
www.babeltech.com
Circle No. 317

Dragon Systems

1-617-925-5200
www.dragonsystems.com
Circle No. 318

Elan

+ 33 5 61 36 89 10
www.elan.fr
Circle No. 319

Fluent Speech Technologies Inc

1-503-748-2110
www.fluent-speech.com
Circle No. 320

GTE, BBN Technologies

1-617-873-4242
www.bbn.com/products/speech
Circle No. 321

IBM Speech and Pen Business Unit

1-800-825-5263
www.ibm.com/viavoice
Circle No. 322

Lernout & Hauspie

1-781-203-5000
www.lhs.com
Circle No. 323

Lucent Technologies

1-630-979-7742
www.lucent.com/speech
Circle No. 324

Natural Microsystems

1-508-620-9300
www.nmss.com
Circle No. 325

Nuance

1-650-847-0000
www.nuance.com
Circle No. 326

Oki Semiconductor

1-408-720-1900
www.okisemi.com
Circle No. 327

Philips Semiconductors

1-408-991-2000
www.semiconductors.philips.com
Circle No. 328

Qualcomm

1-619-658-5005
www.qualcomm.com/cdmatechnologies
Circle No. 329

Radisys

1-503-615-1100
www.radisys.com
Circle No. 330

Sensory

1-408-744-9000
www.sensoryinc.com
Circle No. 331

SpeechWorks

1-617-428-4444
www.speechworks.com
Circle No. 332

THB Germany

0180 3237575
www.thb.de
Circle No. 333

Unisys Natural Language Group

1-610-648-3434
www.unisys.com/marketplace/nlu
Circle No. 334

Vocalis Group plc

+ 44 1223 846177
www.vocalis.com
Circle No. 335

Voice Control Systems

1-972-726-1200
www.voicecontrol.com
Circle No. 336

SUPER CIRCLE NUMBER

For more information on the products available from all of the vendors listed in this box, circle one number on the reader service card. Circle No. 337